Installing Trac as CGI

To install Trac as a CGI script, you need to make the `trac.cgi` executable as a CGI by your web server.

> *Please note that using Trac via CGI is significantly slower than any other deployment method, such as mod_python or FastCGI or even IIS/AJP on Windows.*

If you're using Apache HTTPD, there are a couple ways to do that:

1. Use a `ScriptAlias` to map a URL to the `trac.cgi` script
2. Copy the `trac.cgi` file into the directory for CGI executables used by your web server (commonly named `cgi-bin`). A word of warning, copying the file directly from the repository onto a windows server 2003 machine created difficulties. Rather create a new text file and cut and copy the text into the newly created file. You can also create a symbolic link, but in that case make sure that the `FollowSymLinks` option is enabled for the `cgi-bin` directory.

The first option is recommended as it also allows you to map the CGI to a friendly URL.

Now, edit the Apache configuration file and add this snippet, file names and locations changed to match your installation:

```
ScriptAlias /trac /usr/share/trac/cgi-bin/trac.cgi
```

> *Note that this directive requires the `mod_alias` module to be installed and enabled.*

If you're using Trac with a single project you need to set its location using the `TRAC_ENV` environment variable:

```
<Location "/trac">
  SetEnv TRAC_ENV "/path/to/projectenv"
</Location>
```

Or to use multiple projects you can specify their common parent directory using the `TRAC_ENV_PARENT_DIR` variable:

```
<Location "/trac">
  SetEnv TRAC_ENV_PARENT_DIR "/path/to/project/parent/dir"
</Location>
```

> *Note that the `SetEnv` directive requires the `mod_env` module to be installed and enable. If not, you could set TRAC_ENV in trac.cgi. Just add the following code between "try:" and "from trac.web ...":*

```
import os
os.environ['TRAC_ENV'] = "/path/to/projectenv"
```

> *Or for TRAC_ENV_PARENT_DIR:*

```
import os
os.environ['TRAC_ENV_PARENT_DIR'] = "/path/to/project/parent/dir"
```

This will make Trac available at `http://yourhost.example.org/trac`.

If you are using the Apache suEXEC feature please see http://trac.edgewall.org/wiki/ApacheSuexec.

On some systems, you *may* need to edit the shebang line in the `trac.cgi` file to point to your real Python installation path. On a Windows system you may need to configure Windows to know how to execute a .cgi file (Explorer -> Tools -> Folder Options -> File Types -> CGI).

Mapping Static Resources

Out of the box, Trac will serve static resources such as style sheets or images itself. For a CGI setup, though, this is highly undesirable, because it results in the CGI script being invoked for documents that could be much more efficiently served by the web server directly.

Web servers such as Apache HTTPD allow you to create ?Aliases? to resources, thereby giving them a virtual URL that doesn't necessarily bear any resemblance to the layout of the servers file system. We already used this capability above when defining a `ScriptAlias` for the CGI script, and we'll use it now to map requests to the static resources to the directory on the file system that contains them, thereby bypassing the processing of such requests by the CGI script.

Edit the Apache configuration file again and add the following snippet **before** the `ScriptAlias` for the CGI script , file names and locations changed to match your installation:

```
Alias /trac/chrome/common /usr/share/trac/htdocs
<Directory "/usr/share/trac/htdocs">
  Order allow,deny
  Allow from all
</Directory>
```

Note that whatever URL path you mapped the `trac.cgi` script to, the path `/chrome/common` is the path you have to append to that location to intercept requests to the static resources.

For example, if Trac is mapped to `/cgi-bin/trac.cgi` on your server, the URL of the Alias should be `/cgi-bin/trac.cgi/chrome/common`.

Similarly, if you have static resources in a projects htdocs directory, you can configure apache to serve those resources (again, put this **before** the `ScriptAlias` for the CGI script, and adjust names and locations to match your installation):

```
Alias /trac/chrome/site /path/to/projectenv/htdocs
<Directory "/path/to/projectenv/htdocs">
  Order allow,deny
  Allow from all
</Directory>
```

Alternatively, you can set the `htdocs_location` configuration option in trac.ini:

```
[trac]
htdocs_location = /trac-htdocs
```

Trac will then use this URL when embedding static resources into HTML pages. Of course, you still need to make the Trac `htdocs` directory available through the web server at the specified URL, for example by copying (or linking) the directory into the document root of the web server:

```
$ ln -s /usr/share/trac/htdocs /var/www/your_site.com/htdocs/trac-htdocs
```

Note that in order to get this `htdocs` directory, you need first to extract the relevant Trac resources using the `deploy` command of TracAdmin:

```
deploy <directory>
        -- Extract static resources from Trac and all plugins.
```

Adding Authentication

The simplest way to enable authentication with Apache is to create a password file. Use the `htpasswd` program to create the password file:

```
$ htpasswd -c /somewhere/trac.htpasswd admin
New password: <type password>
Re-type new password: <type password again>
Adding password for user admin
```

After the first user, you dont need the "-c" option anymore:

```
$ htpasswd /somewhere/trac.htpasswd john
New password: <type password>
Re-type new password: <type password again>
Adding password for user john
```

See the man page for `htpasswd` for full documentation.

After you've created the users, you can set their permissions using TracPermissions.

Now, you'll need to enable authentication against the password file in the Apache configuration:

```
<Location "/trac/login">
  AuthType Basic
  AuthName "Trac"
  AuthUserFile /somewhere/trac.htpasswd
  Require valid-user
</Location>
```

If you're hosting multiple projects you can use the same password file for all of them:

```
<LocationMatch "/trac/[^/]+/login">
  AuthType Basic
  AuthName "Trac"
  AuthUserFile /somewhere/trac.htpasswd
  Require valid-user
</LocationMatch>
```

For better security, it is recommended that you either enable SSL or at least use the ?digest? authentication scheme instead of ?Basic?. Please read the Apache HTTPD documentation to find out more. For example, on a Debian 4.0r1 (etch) system the relevant section in apache configuration can look like this:

```
<Location "/trac/login">
    LoadModule auth_digest_module /usr/lib/apache2/modules/mod_auth_digest.so
```

```
    AuthType Digest
    AuthName "trac"
    AuthDigestDomain /trac
    AuthUserFile /somewhere/trac.htpasswd
    Require valid-user
</Location>
```

and you'll have to create your .htpasswd file with htdigest instead of htpasswd as follows:

```
# htdigest /somewhere/trac.htpasswd trac admin
```

where the "trac" parameter above is the same as AuthName above ("Realm" in apache-docs).

See also: TracGuide, TracInstall, TracFastCgi, TracModPython